MULTIMEDIA UNIVERSITY

STUDENT ID NO

□□□□□□□□□□

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2019/2020

TCP2451 – PROGRAMMING LANGUAGE TRANSLATION

(All Sections / Groups)

28 February 2020
03:00 PM– 05:00 PM
(2 Hours)

INSTRUCTION TO STUDENT

1. Answer **ALL** questions.
2. This question paper has 10 printed pages excluding the front cover.
3. Print all your answers **CLEARLY** in the specific answer box provided for each question.
4. Submit this question paper at the end of the examination.

QUESTION 1 [25 marks]

a) A C-style variable name contains letters, digits, and underscores. The variable name must start with a letter and may not contain any spaces.

Write a regular expression that will accept a valid variable names. **[4 marks]**

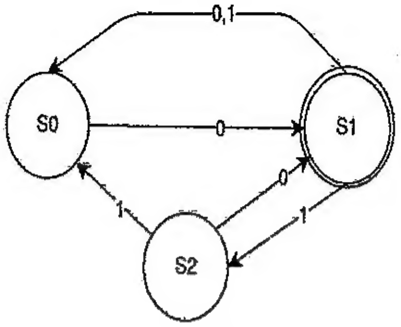
b) Construct a Non-Deterministic-Finite Automaton (NFA) using the Thompson construction method that is able to recognize the sentences generated by the regular expression: RE = (ab)*.(a)*

[6 marks]

c) Identify the inputs and the outputs for the Lexical and Syntax analysis phases of a compiler. **[6 marks]**

Continued ...

d) Given the Finite Automaton below with initial state 0 and alphabets {a,b} answer the following questions (1) and (2):



1. Why is this FA a Non-Deterministic Finite Automaton (NFA)?
2. Convert this NFA to a DFA using set construction method.

[9 Marks]

Continued ....

QUESTION 2 [25 marks]

Given the following Context Free Grammar G={S, {S, U,V,W},{a,b,c,d}, P} with P as shown below:

$$
\begin{aligned}
S &\rightarrow UVW \\
U &\rightarrow (S) \mid aSb \mid d \\
V &\rightarrow aV \mid \varepsilon \\
W &\rightarrow cW \mid \varepsilon
\end{aligned}
$$

(a) Write the FIRST and FOLLOW sets for all the non-terminals in this grammar.

[6 Marks]

	First	Follow
S		
U		
V		
W		

(b) Using the LL(1) table, show how a table-driven parser parses the string "caca"

State	a	c	$
S		S→cL	
L	L→aL	L→cL	L→ε

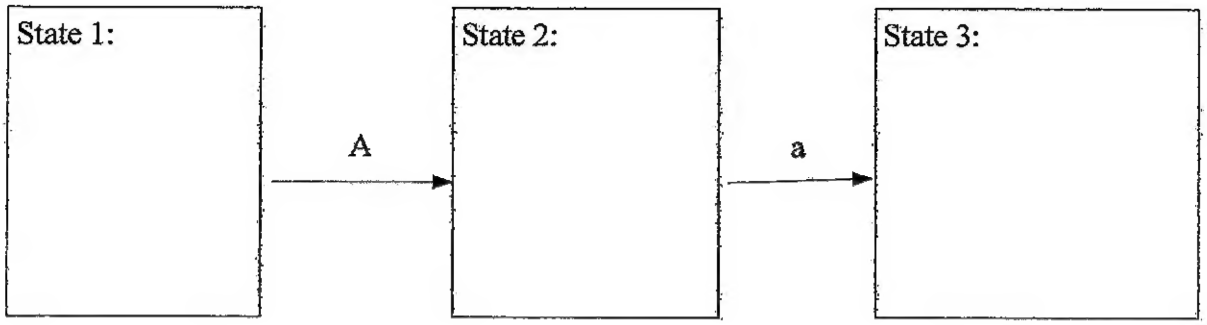
[6 Marks]

Stack	Input	Action

Continued ...

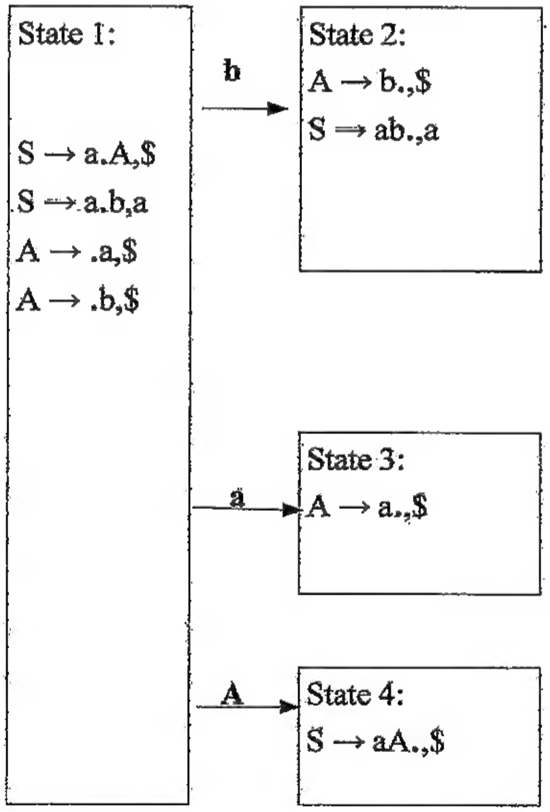
(c) Derive the canonical LR(1) items for the augmented grammar below: **[6 Marks]**

S → A
A → Aa
A → ε



(d) Given the following sets of LR(1) items, construct the corresponding entries in the LR(1) parse table by filling the table below: **[7 Marks]**

LR(1) Items



LR(1) Parse Table

State	Action			Go to	
	a	b	$	S	A

Continued ...

QUESTION 3 [25 marks]

(a) Consider the following grammar:

$$S \rightarrow ABd$$
$$A \rightarrow aA \mid \varepsilon$$
$$B \rightarrow b \mid cA$$

Construct its recursive-descent parser (with lookahead), given the following functions:

[8 Marks]

```
lookahead;        // current token
match(t) {        // matches token
  if (lookahead== t)    // found match
     lookahead= next_token();  // get next
  else
     error();    // else error
}

 parser(){
      lookahead = next_token();  // init
      S( );                      // start symbol
      match("$");                // match EOF
}
```

Continued ...

(b) Write a Lex specification file to generate a lexical analyzer to identify two different types of inputs namely, integer and float. An input of q or Q will cause the lexer to terminate. Any other input is treated as an error.
The lexer will display “valid integer” if the input is an integer value, “valid float” if the input is a float number; and “error” otherwise.
Use the following regular expressions in your lex file.

RE for Integers = [\t]*[-+]?[0-9]*
RE for floats = [\t]*[-+]?[0-9]*"\."[0-9]+
RE for q or Q = [\t]*[qQ]
RE for anything else = .+

[9 Marks]

Continued ...

(c) Given the following expression:

$$a + a * (b - c) + (b - c) * d$$

(1) Generate the three address intermediate code for the expression

[4 Marks]

(2) Give the triples representation for the intermediate code generated in (1)

[4 Marks]

Continued ...

QUESTION 4 [25 marks]

(a) Given the following code:

```
1:          a = b
2:  L1:     b = c
3:  L2:     if (...) goto L4
4:          c = b
5:  L3:     d = a
6:  L4:     goto L1
7:  L5:     b = a
8:  L6:     if (...) goto L2
9:          c = a
```

(1) Identify the basic blocks and draw the control flow graph.

[7 Marks]

(2) What is the compiler term used to describe the statements 7, 8, and 9.

[2 Marks]

Continued ...

(b) Consider the basic block given below:

```
a = b + c
b = b + c
c = B + c
```

Draw the Directed Acyclic Graph that represents the basic block and explain how the DAG can be used to help optimize the given code.

[5 marks]

(c) Explain what is type checking with respect to programming language translation, and Explain the difference between static and dynamic type checking.

[6 Marks]

Continued ...

(c) Identify the Inherited and Synthesized attributes in the the following attributed grammar. Explain your decision for each attribute.

Production Rules	Semantic Attributes

Number → Sign List	List.pos ← 0
	if Sign.neg then Number.val ← -List.val
	else Number.val ← List.val

Sign → +	Sign.neg ← false
Sign → -	Sign.neg ← true

List → List , Bit	List1.pos ← List0.pos + 1
	Bit.pos ← List0.pos
	List0.val ← List1.val + Bit.val

List → Bit	Bit.pos ← List.pos
	List.val ← Bit.val

Bit → 0	Bit.val = 0
Bit → 1	Bit.val = $2^{\text{Bit.pos}}$

[5 Marks]

End of Page